Argonne
NATIONAL
LABORATORY

... for a brighter future

The ROMIO-HDF5 Interplay

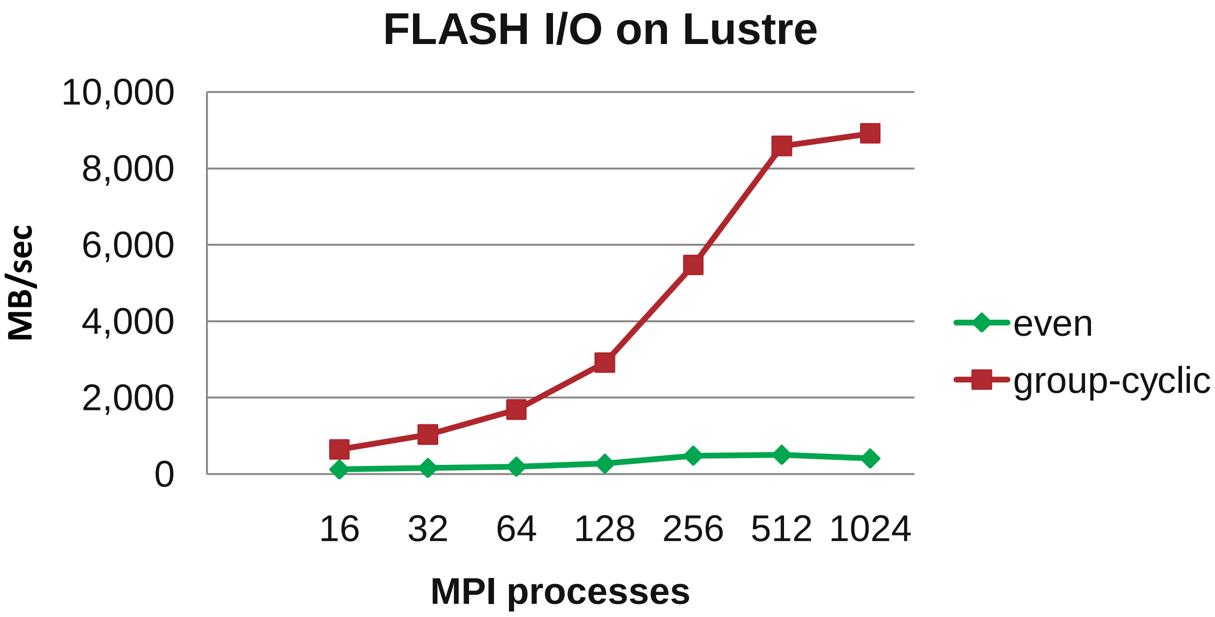
Rob Latham

robl@mcs.anl.gov

Mathematics and Computer Science Division

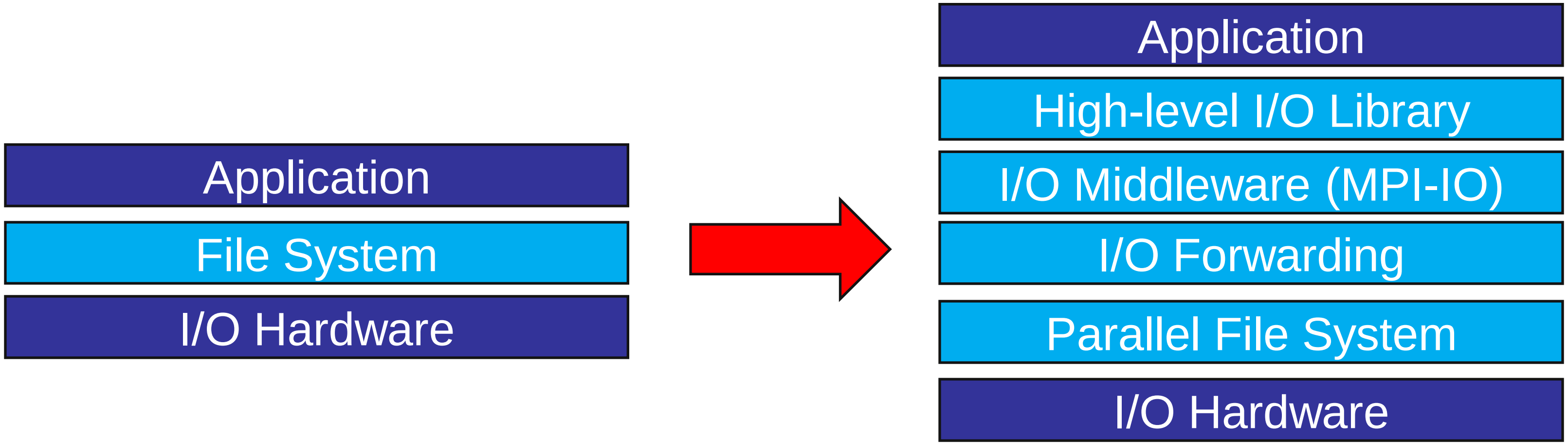
Argonne National Laboratory

MPI-IO: Maybe not as broken as believed?



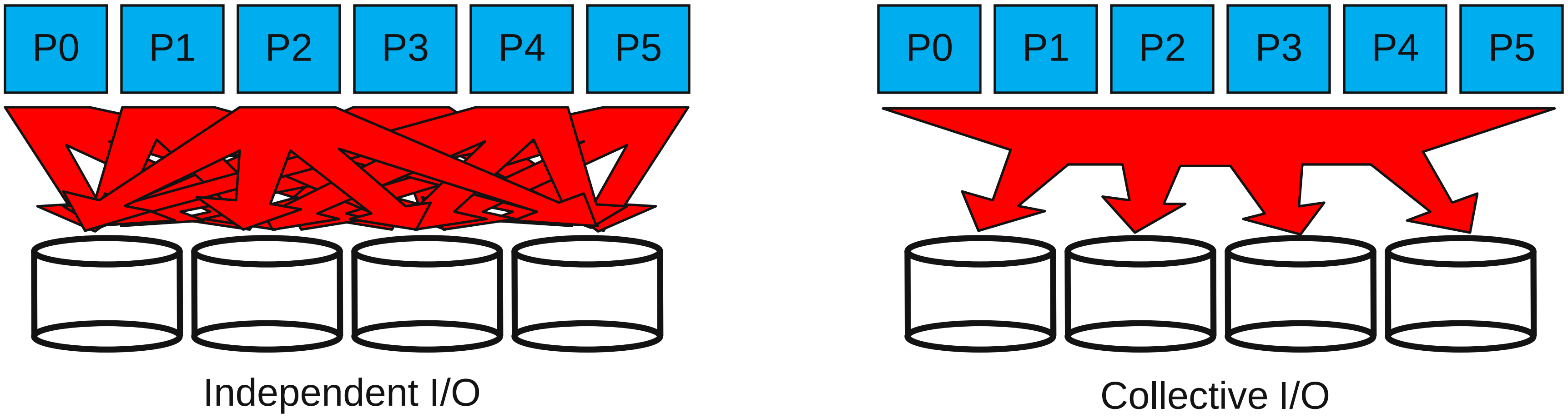
- Wei-keng Liao, Northwestern University, SC2009
  - More detail later

Background: Software for Parallel I/O in HPC



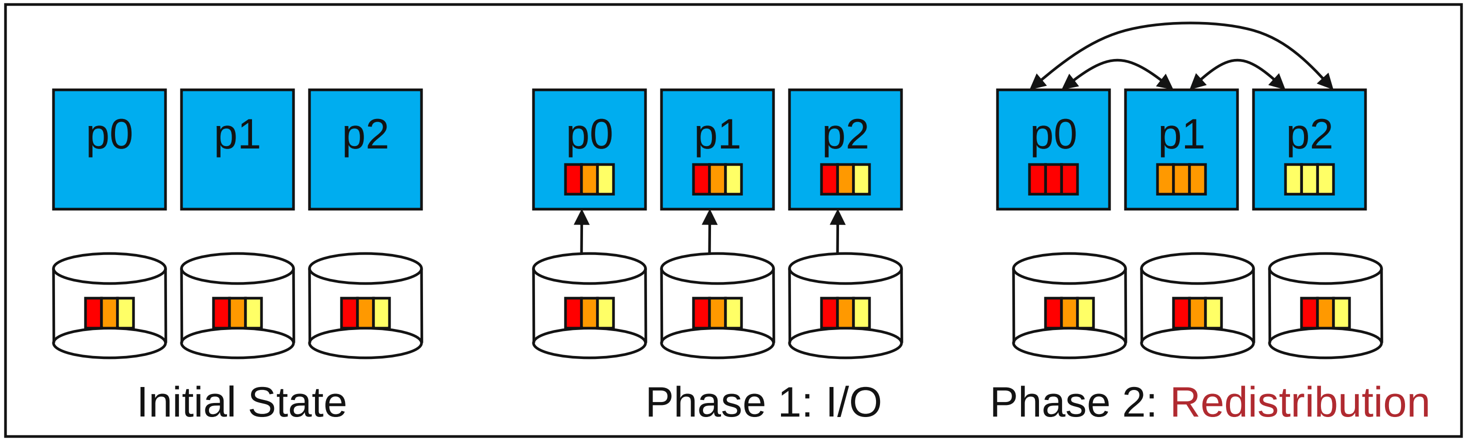
- Applications require more software than just a parallel file system
- Support provided via multiple layers with distinct roles:
  - Parallel file system maintains logical space, provides efficient access to data (e.g. PVFS, GPFS, Lustre)
  - I/O Forwarding found on largest systems to assist with I/O scalability
  - Middleware layer deals with organizing access by many processes (e.g. MPI-IO, UPC-IO)
  - High level I/O library maps app. abstractions to a structured, portable file format (e.g. HDF5, Parallel netCDF)
- Goals: scalability, parallelism (high bandwidth), and usability

Independent and Collective I/O



- Independent I/O operations specify only what a single process will do
  - Independent I/O calls do not pass on relationships between I/O on other processes
- Many applications have phases of computation and I/O
  - During I/O phases, all processes read/write data
  - We can say they are collectively accessing storage
- Collective I/O is coordinated access to storage by a group of processes
  - Collective I/O functions are called by all processes participating in I/O
  - Allows I/O layers to know more about access as a whole, more opportunities for optimization in lower software layers, better performance

Collective I/O and Two-Phase I/O

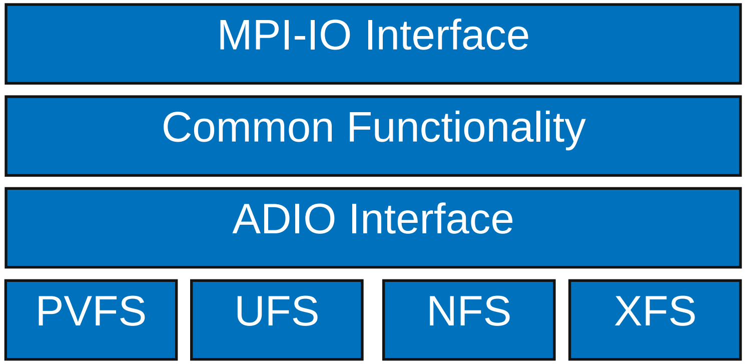


Two-Phase Read Algorithm

- Problems with independent, noncontiguous access
  - Lots of small accesses
  - Independent data sieving reads lots of extra data, can exhibit false sharing
- Idea: Reorganize access to match layout on disks
  - Single processes use data sieving to get data for many
  - Often reduces total I/O through sharing of common blocks
- Second "phase" redistributes data to final destinations
- Two-phase writes operate in reverse (redistribute then I/O)
  - Typically read/modify/write (like data sieving)
  - Overhead is lower than independent access because there is little or no false sharing
- Note that two-phase is usually applied to file regions, not to actual blocks

ROMIO MPI-IO Implementation

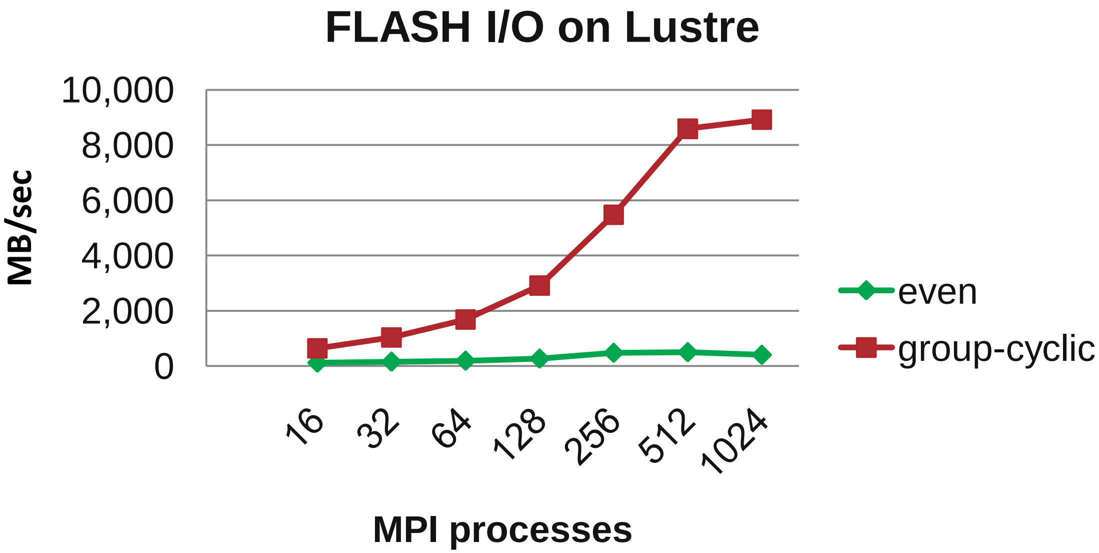
- Developed at Argonne National Laboratory
  - Leverages MPI-1 communication
  - Supports local file systems, network file systems, parallel file systems
    - *UFS module works for GPFS, Lustre, and others*
    - *Tuned modules for PVFS, BlueGene. Lustre in development*
  - Includes data sieving and two-phase optimizations
- Basis for several vendor implementations
  - IBM BlueGene/L, BlueGene/P
  - Cray XT3/XT4/XT5
  - MPICH2
  - OpenMPI



ROMIO's layered architecture.

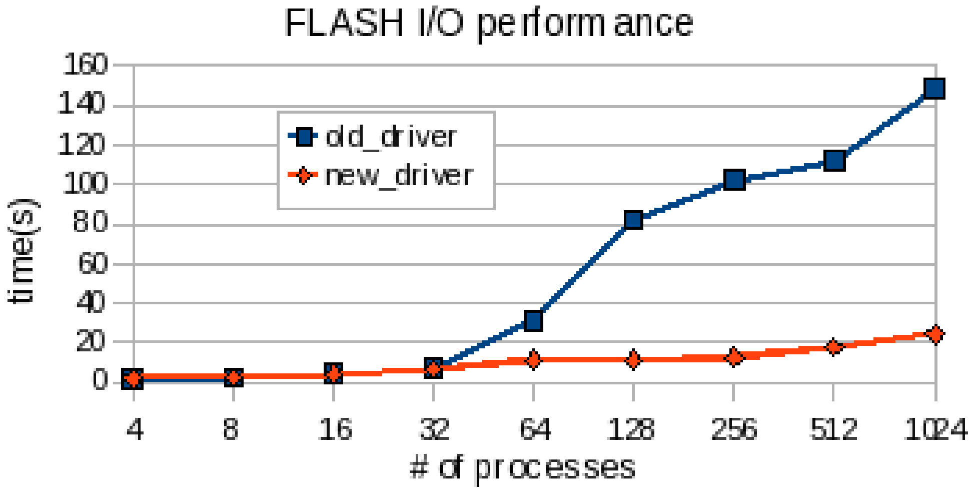
ROMIO improvements: #1 – Flexible redistribution

- Wei-keng Liao: SC 2008
- New two-phase (ADIO-level) decompositions:
  - Even Alignment: file system lock boundaries
    - *Best for GPFS*
  - Static-cyclic: distribute relative to I/O servers
    - *Interacts poorly with readahead*
  - Group-cyclic: grouping based on I/O servers, distribute among group
    - *Best for Lustre writes*



ROMIO Improvements: #2 – File-system specific tuning

- Weikuan Yu (Oak Ridge), Emoly Liu (Sun): Lustre-specific driver for ROMIO
  - Thresholds for disabling collective I/O
  - Thresholds for disabling data sieving
  - New hints (MPI-IO tuning parameters)
  - Lustre-specific ioctl() commands



Time to write checkpoint: Smaller is better.

ROMIO Improvements: #3 – Site-specific hints

- MPI Info parameters
  - Flexible: string-based keyword-value pairs
  - Portable: implementations free to ignore hints they don’t understand
  - Rarely used
- MPICH2-1.0.7: ROMIO can read hints from config file
  - /etc/romio-hints or ROMIO_HINTS env var
  - E.g. good place for Lustre-specific knobs

HDF5 and MPI-IO

- Good: trust MPI-IO library
  - It’s usually ROMIO
  - Describe I/O with MPI datatypes
  - Use file format layout information
  - Use collective I/O
- Bad:
  - Filesystem-specific optimizations
    - *Yes, I know there are a few in there already*
  - System-specific optimizations
    - *Better served in MPI-IO layer*
  - Reinventing anything MPI-IO folks already did

Research vs. Production

"System software research is irrelevant" - Rob Pike, 2000

- False, of course
- Argonne, Northwestern University, Oak Ridge, others continue to develop, refine ROMIO
- Vendors incorporate research... at varying rates
- Good: IBM and BGP
  - ANL developed PVFS enhancements
  - IBM incorporates into production driver; installed 3 months later
- Bad: Cray and XT
  - As of summer 2008, Franklin's MPI-IO based on MPICH2-1.0.4
  - MPICH2-1.0.4 released August, 2006

"Best" way to get ROMIO research onto production machines?